Science For A Better Life

Flashback Technologies

Back to the future Part 1

2012-09-01 Andreas Stephan, Bayer Business Services GmbH

Agenda

- **The princess**
- The bad guy
- The hero
- The rescue of the beautiful princess
- And if they didn´t delete, they….

The princess

- Data in DBs make only sense if consistent and correct
- Datamodel and/or business model define and implement
  - Consistency (referential constraints, transaction-logic)
  - Correctness (e.g. Check-Constraints, Trigger-Logic, …)
- Transition from one consistent state to the other
  - Apply business rules and datamodel logic
  - Changes have to
    - Respect referential and check-constraints
    - Run or accept transaction- and trigger-logic
    - Allow for „smart“ terms (retention cycles, legal storage cycles, etc.)

Agenda

- The princess
- **The bad guy**
- The hero
- The rescue of the beautiful princess
- And if they didn´t delete, they….

The bad guy

- Manually overriding business logic and datamodel
  - Change consistency and correctness of data
  - May create consistent but wrong data
    - E.g. deleting details rows of master-detail tables (customer, orders)
    - E.g. updates without where clause
- Avoiding of unwanted modifications not always possible
  - SW-Updates (are all business rules implemented correctly?
  - Migration scenarios (possibly with disabling constraints/triggers)
  - Unwanted or faulty operator or DBA activity
  - SW faults
- Queries and reports based on an exact time only possible with temporary tables and jobmanagement without flashback

Agenda

- The princess
- The bad guy
- **The hero**
- The rescue of the beautiful princess
- And if they didn´t delete, they….

The hero

- Flashback introduced with Oracle 9i as a „waste product“ of UNDO management
  - Select * from tabelle AS OF TIMESTAMP ….;
  - Select * from tabelle AS OF SCN …;
- Flashback limited through UNDO TS (size and retention parameter)
  - Hours up to few days are realistic
- Extended features introduced with 10g and 11g
- Mixing up of terms and functions – e.g Flashback / RecycleBin both called flashback, but are completely different technologies
  - Drop Table stores in Data TS instead of UNDO TS
  - Flashback table to before drop renames dropped table (bin$..)

Flashback methods

- Flashback Query
- Flashback Version Query
- Flashback Transaction Query
- DBMS_FLASHBACK Package
- Flashback Transaction
- Flashback Data Archive (Total Recall)

- Flashback Table
- Flashback Drop
- Flashback Database

Flashing back

Prerequisites

- UNDO Tablespace large enough to hold all logging information
- Retention parameter set accordingly
- Enable supplemental logging
  - Needed for “Flashback transaction” feature
  - Minimal supplemental logging
    - Alter database add supplemental log data;
  - Primary key supplemental logging
    - Alter database add supplemental log data (primary key) columns;
  - Foreign key supplemental logging
    - Alter database add supplemental log data (foreign key) columns;
    - many foreign key constraints → performance penalty

Flashing back

- Caveats
  - DDL statements which change the structure of a table
    - Drop/modify columns
    - Move table
    - Drop partition
    - Truncate table/partition
    - Add constraint

    ➔ Invalidates the UNDO data for this table! ORA-01466 will occur
  - Flashback will not work on v$ Views
  - Flashback will work on static dictionary objects (sometimes ☹)
  - Don´t mix up static with dynamic dictionary objects
    SQL> select to_char(sysdate,'DD.MM.YYYY HH24:MI:SS') from dual as of timestamp sysdate-5;

Agenda

- The princess
- The bad guy
- The hero
- **The rescue of the beautiful princess**
- And if they didn´t delete, they….

Simple Flashback Query

- Query time based or SCN based
  - to_date 1s exact, to_timestamp 1/1000s
  - Flashback on timestamp exact within +/- 3 seconds
  - Flashback on SCN is exact
  - Select princess_happy from kings_castle as of timestamp
    to_date('DD.MM.YYYY HH24:MI:SS','01.01.1989 11:39:41');
  - Select queen_happy from kings_castle as of timestamp
    to_timestamp('DDMMYYYY HH24:MI:SS','01.01.1989 11:39:41');
  - Get current SCN
    select dbms_flashback.get_system_change_number from dual;
  - Select king_happy from kings_castle as of SCN 12345;
- UNDO Size and Retention make this query work (or not)

Even simpler Flashback Query

- Login at 9am
- SQL> connect arnie/blackenedger
  SQL> set transaction read only;
- Go away for 8hours
- Run any SQL query and it will use the values of 9am !
  (OK, as far as the RBS or UNDO can handle the data)

- This works for every RDBMS version since 5.x !!
  (Multi version read consistency)
- So, Flashback in 7.3 works

Simple flashback query - Praxis

Counting the number of rows created in the past hour(s)

```
SQL>  select count(t2.cust_id)-count(t1.cust_id)
        from flashtest1 t1 as of timestamp(sysdate-1/24), flashtest1 t2;
```

doesn´t work because alias not supported when using „as of“ clause!

Solution1: create views with realtime and relative delay contents and query the diff

```
SQL>create view flashtest1_rt AS select count(*) as flcount from flashtest1;

SQL>create view flashtest1_1hdelay AS select count(*) as flcount
        from flashtest1 as of timestamp (SYSTIMESTAMP - INTERVAL '60' MINUTE);

SQL> select t1.flcount - t2.flcount from flashtest1_rt t1, flashtest1_1hdelay t2;
```

or easier&flexible Solution 2:

```
SQL> with
        q1 as (select count(*) as s1 from flashtest1
                as of timestamp (SYSTIMESTAMP - INTERVAL '60' MINUTE)),
        q2 as (select count(*) as s2 from flashtest1)
        select s2-s1 as "Diff" from q2,q1;
```

Flashback - Exactness

- Time based flashback has a resolution of +/- 3 seconds
  - Historic cause: smon_scn_time stores scn-to-time records on a 5 minute interval
  - Each interval has 100 entry slots (5*60 = 300 seconds)
    → each slot has a bandwidth (=inaccuracy) of 3 seconds
  - Historic: Versions up to 10g allowed only 1440 rows in the smon_scn_time table (circular buffer – overwrite oldest row after 1440 rows are reached)
    - 1440*5Minutes = 7200 minutes = 120 hours = 5 days
    - Only flashback over 5 days (maximum) will work
    - Shutdown the db for one year and bring it up again
      → flashback queries up to 1year+5days will work!
- Flashing back using SCNs is exact!

Flashback version query

- Show all changes to rows based on the information in the UNDO segments
- SQL> select cust_id, cust_name, last_sale, last_amount, sum_to_date, last_change,
  last_scn, versions_starttime, versions_endtime
  from flashtest1
  versions between scn 6796000 and 6797093
  -- or – versions between timestamp to_date… and timestamp to_date…
  where cust_id=1000;

```
CUST_ID    CUST_NAME            LAST_SALE           LAST_AMOUNT SUM_TO_DATE LAST_CHANGE
LAST_SCN   VERSIONS_STARTTIME   VERSIONS_ENDTIME
---------- -------------------- ------------------- ----------- ----------- -------------------
1000       Oracle               24.09.2012 11:02:32     2000.00     5345.00 24.09.2012 11:02:32
6797091    24.09.2012 11:02:34

1000       Oracle               24.09.2012 10:37:49     1000.00     3345.00 24.09.2012 10:37:49
6796086    24.09.2012 11:00:46  24.09.2012 11:02:34

1000       Oracle               12.09.2012 15:52:43     2345.00     2345.00 12.09.2012 15:52:43
                                24.09.2012 11:00:46
```

Flashback Transaction Query

- Erroneously changed data? Need to roll back?
  - Flashback transaction query shows what, when, who and the corresponding UNDO SQL!
  - Needs Supplemental logging (data, primary and ideally foreign key)
    - Beware of performance problems if enabling supplemental logging for foreign key constraints on heavily modified tables!
  - Example:

```
SQL> select to_char(start_timestamp,'DD.MM.YYYY H24:MI:SS'),
            logon_user,operation,table_name,table_owner,row_id,undo_sql
       from flashback_transaction_query
       where table_name like 'FLASH%,
          and operation != 'UNKNOWN';
```

  - Use these UNDO SQLs with extreme care. They don´t know (and don´t care) about triggers or other logic on the tables to be modified. This may garble all your data!

Flashback Transaction Query

```
START_TIMES         LOGON_USER           OPERATION                        TABLE_NAME                     TABLE_OWNER                      ROW_ID
------------------- -------------------- -------------------------------- ------------------------------ -------------------------------- --------
UNDO_SQL
----------------------------------------------------------------------------------------------------------------------------------------------------
24.09.2012 15:57:19 FLTEST               INSERT                           FLASHTEST1                     FLTEST                           AAAGByAA
delete from "FLTEST"."FLASHTEST1" where ROWID = 'AAAGByAAFAAAACbAAG';

24.09.2012 15:55:13 FLTEST               UPDATE                           FLASHTEST1                     FLTEST                           AAAGByAA
update "FLTEST"."FLASHTEST1" set "LAST_SALE" = NULL, "LAST_AMOUNT" = NULL, "SUM_TO_DATE" = NULL, "LAST_CHANGE" = TO_DATE('20-SEP-12', 'DD-MON-RR')

24.09.2012 15:55:13 FLTEST               UPDATE                           FLASHTEST1                     FLTEST                           AAAGByAA
update "FLTEST"."FLASHTEST1" set "LAST_SALE" = NULL, "LAST_AMOUNT" = NULL, "SUM_TO_DATE" = NULL, "LAST_CHANGE" = TO_DATE('20-SEP-12', 'DD-MON-RR')

24.09.2012 15:55:13 FLTEST               UPDATE                           FLASHTEST1                     FLTEST                           AAAGByAA
update "FLTEST"."FLASHTEST1" set "LAST_SALE" = NULL, "LAST_AMOUNT" = NULL, "SUM_TO_DATE" = NULL, "LAST_CHANGE" = TO_DATE('20-SEP-12', 'DD-MON-RR')
```

Flashback Table (if dropped)

- Dropped tables aren´t really dropped starting with 10g but „moved“ to the recyclebin – except you didn´t want this behaviour (by de-configuring it)
- This feature is called „flashback table“ but has nothing to do with UNDO and all the other flashback stuff.
  - Renames tables to bin$xxxxxxx instead of dropping
  - Some logic checking bin$ entries purging old entries if space is needed
  - Can be switched back to old behaviour by db parameter
  - May be a real problem if application logic contains lots of drop table SQLs
  - May be a real life-saver if you dropped a critical table
    - Recover using „flashback table to before drop“ needs seconds
    - Recover using traditional clone database from backup needs hours or days

Flashback Table (if dropped)

- Multi-version „undrop“ is possible if frequent drop/create actions exist
  - Flashback table test to before drop; -- this brings version-1 to life
  - Rename table test to test01;
  - Flashback table test to before drop; -- this brings version-2 to life
  - Rename table test to test02;
  - Flashback table test to before drop; -- this brings version-3 to life
  - … and so on until space is exhausted and Oracle decided to purge older bin$ entries for this table
  - The tables could be of completely different structure/content (different columns/constraints etc.)

Flashback Table (to time/to scn)

- If the table wasn´t dropped but values modified/deleted you may flashback the whole table
- This feature again uses the UNDO (<> flashback table to before drop)
- Prerequisite: row movement on table must be enabled
  - Alter table flashtest1 enable row movement;
- Flashback table flashtest1 to scn xxxxxxx;
- Flashback table flashtest1 to timestamp sysdate-1;

Flashback database

- If all or too many tables are wrong
  - Switch back the whole database without restoring from backup
- Prerequisites
  - Enough free space in flashback area (db_recovery_file_dest & _size)
    Again no usage of UNDO!
  - Parameter db_flashback_retention_target
  - Alter database flashback on; (while db is in mounted state)
  - restore point with guarantee
    - create restore point last_line_of_defense guarantee flashback database;
- Flashback database must be called as sysdba in mount state
  - Flashback database to restore point last_line_of_defense;
    Alter database open resetlogs;

Exporting data based on time/scn

- Export data from a known time or scn using standard exp or datapump
  - exp userid=fltest full=Y flashback_scn=7133041 file=flash_exp_01.dmp
  - expdp userid=fltest dumpfile=fltest.dmp flashback_time=sysdate-8

- This feature uses the UNDO again, so errors will occur if blocks aren´t stored in UNDO any more (e.g. after retention timed out)
  - EXP-00008: ORACLE error 1555 encountered
    ORA-01555: snapshot too old: rollback segment number 6 with name "_SYSSMU6_2758131390$" too small
  - ORA-02354: error in exporting/importing data
    ORA-01555: snapshot too old: rollback segment number 3 with name "_SYSSMU3_976774123$" too small

Transaction backout w/ dependencies

- If a transaction consists of several statements (as usual)
  or
  a transaction deletes master/details rows using „on delete cascade“
- Normal single table flashback wouldn´t be sufficient
  - dbms_flashback.transaction_backout package could solve the problem
- First look into special columns from flashback_transaction_query
  - select versions_startscn startscnm, versions_endscn endscn, versions_xid xid, versions_operation oper, cust_id, order_id from flashtest_detail versions between scn minvalue and maxvalue;
  - Next look up which tables/operations where involved
  - select start_scn, commit_scn, logon_user, operation, table_name, undo_sql from flashback_transaction_query
    where xid= hextoraw(value_from_above);

Transaction backout w/ dependencies

- Now you can call the package (as SYS!)
- ```
  SQL> begin
  2  dbms_flashback.transaction_backout
  3
  4  (
  5   numtxns => 1 ,
  6  xids => xid_array('03001E006D130000'),
  7   options => DBMS_FLASHBACK.CASCADE
  8  );
  9  end;
  ```
- ```
  ERROR at line 1:
  ORA-55511: Flashback Transaction experienced error in executing undo SQL
  ORA-02291: integrity constraint
  (FLTEST.FLASHTEST_DETAIL_FLASHTES_FK1) violated - parent key not found
  ```

  Oracle doesn´t care about constraints and starts actions in the wrong order (if any).

Transaction backout w/ dependencies

- These steps have to be run in addition
  - Disable Constraints
  - Call package
  - Commit data
  - Enable Constraints

- Be extremely cautious when using this feature
- There´s no guarantee that it works
- High risk that you shred your data
- One should have very good knowledge of structure, dependencies, constraints etc. when planning to use transaction backout!

And what about BLOBs?

- Before-Images of BLOBs or CLOBs aren´t stored in UNDO Segments but in the BLOB itself
- Time to flash back depends on version / retention parameter
  - ```
    CREATE TABLE FLTEST.FLASHBLOB3 (BLOBL_CONTENT CLOB)
    LOB (BLOBL_CONTENT) STORE AS BASICFILE (TABLESPACE USERS DISABLE
    STORAGE IN ROW RETENTION NOCACHE LOGGING ) ;
    ```
  - ```
    Begin
      insert into flashblob3 values (utl_raw.cast_to_raw('BLOB inserted on
       '||to_char(sysdate,'DD.MM.YYYY HH24:MI:SS')||' with
       old_scn='||to_char(dbms_flashback.get_system_change_number)));
      commit;
      dbms_lock.sleep(1); -- sleep 1 second
      …
    select utl_raw.cast_to_varchar2(dbms_lob.substr(blobl_content)) as
    "blob2_content" from flashblob3;
    ----------------------------------------------------------------
    ------------------
    BLOB inserted on 11.10.2012 08:51:39 with old_scn=7788696
    ```

And what about BLOBs?

- Same 3 second variation when using „as of timestamp“ on BLOBs
- SCN queries are exact (as for normal tables/columns)

- In fact, SCN to timestamp variation doesn´t depend on type of column
- `SQL>select scn_to_timestamp(7805381) from dual`
  `11-OCT-12 04.36.20.000000000 PM`
- `SQL>select scn_to_timestamp(7805389) from dual;`
  `11-OCT-12 04.36.20.000000000 PM`
- `SQL>select scn_to_timestamp(7805391) from dual`
  `11-OCT-12 04.36.23.000000000 PM`

Total Recall (!! License required!!)

- Another complete – and fully new – solution for archiving old versions of rows
- Change information is stored in a (dedicated) tablespace
  - `SQL> CREATE TABLESPACE ARNIE DATAFILE 'arnie_01.dbf' SIZE 100M;`
- Retention time is stated in granularity of years or months
  - `SQL> CREATE FLASHBACK ARCHIVE schwarzenegger TABLESPACE arnie RETENTION 1 YEAR; -- privilege “flashback archive administer” required!`
- Activate flashback for a table
  - `SQL> grant flashback archive on schwarzenegger to fltest;`
    `SQL> alter table flashtest1 flashback archive schwarzenegger;`
- Queries work like normal flashback queries, Oracle transparently accesses the data in the flashback archive
  - Even SCN queries should work – although it makes few sense to query data one year old on a SCN granularity
  - Deleted rows (from the original table) are accessible from the flashback archive

Agenda

- The princess
- The bad guy
- The hero
- The rescue of the beautiful princess
- **And if they didn´t delete, they….**

…they´ll be consistent up to date!

Q & A

Science For A Better Life

Thank you!

2012-09-01 Andreas Stephan, Bayer Business Services GmbH

Forward-Looking Statements

This presentation may contain forward-looking statements based on current assumptions and forecasts made by Bayer Group or subgroup management.

Various known and unknown risks, uncertainties and other factors could lead to material differences between the actual future results, financial situation, development or performance of the company and the estimates given here. These factors include those discussed in Bayer's public reports which are available on the Bayer website at www.bayer.com.

The company assumes no liability whatsoever to update these forward-looking statements or to conform them to future events or developments.

Zukunftsgerichtete Aussagen

Diese Präsentation kann bestimmte in die Zukunft gerichtete Aussagen enthalten, die auf den gegenwärtigen Annahmen und Prognosen der Unternehmensleitung des Bayer-Konzerns bzw. seiner Teilkonzerne beruhen.

Verschiedene bekannte wie auch unbekannte Risiken, Ungewissheiten und andere Faktoren können dazu führen, dass die tatsächlichen Ergebnisse, die Finanzlage, die Entwicklung oder die Performance der Gesellschaft wesentlich von den hier gegebenen Einschätzungen abweichen. Diese Faktoren schließen diejenigen ein, die Bayer in veröffentlichten Berichten beschrieben hat. Diese Berichte stehen auf der Bayer-Webseite www.bayer.de zur Verfügung.

Die Gesellschaft übernimmt keinerlei Verpflichtung, solche zukunftsgerichteten Aussagen fortzuschreiben und an zukünftige Ereignisse oder Entwicklungen anzupassen.